Example:

Implementing alias_method_chain

In this task, we will create own method that imitates Rails' "alias_method_chain" method. Also, this task will show you how you can create methods that change the class behaviour or definition of the classes where they were called.

Background: alias_method_chain

```ruby
class MyClass
    def foo
        "original foo"
    end

    def foo_with_extra
        "#{foo_without_extra} - foo with extra"
    end

    alias_method_chain :foo, :extra

    # foo refers to `foo_with_extra`
    # original `foo` refers to `foo_without_extra`
end

obj = MyClass.new

obj.foo                                 # "original foo - foo with extra"
obj.foo_without_extra                   # "original foo"
```

Task:

Assume we have “foo” and “foo_with_extra” methods defined. After running the following code within the class definition:

```
alias_method_chain :foo, :extra
```

The “foo” method must refer to “foo_with_extra” method. And, original implementation of “foo” method must be still available in “foo_without_extra” method.

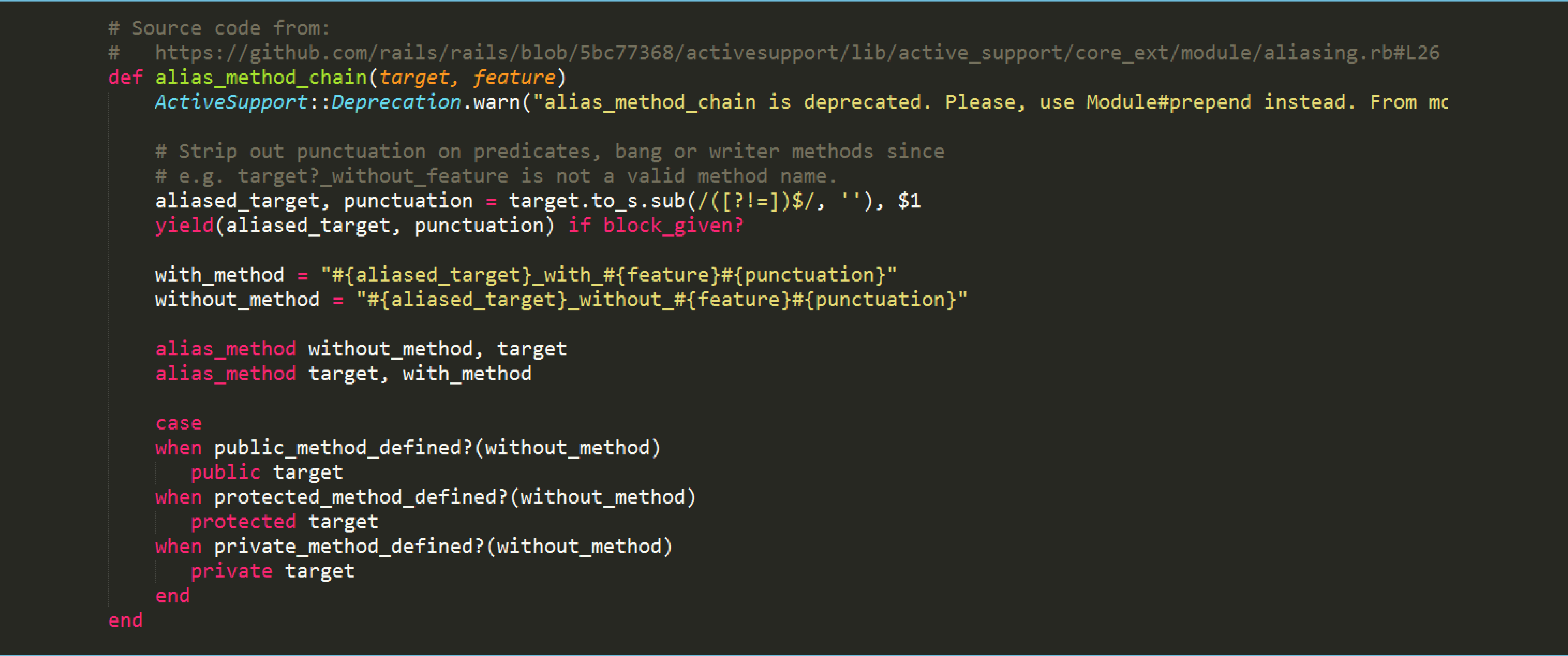
Use alias_method to create new aliases

```ruby
class Module
    def alias_method_chain_imitator(original, extra)
        without_extra = "#{original}_without_#{extra}"
        alias_method without_extra, original
        with_extra    = "#{original}_with_#{extra}"
        alias_method original, with_extra
    end
    private :alias_method_chain_imitator
end
```

Use alias_method to create new aliases

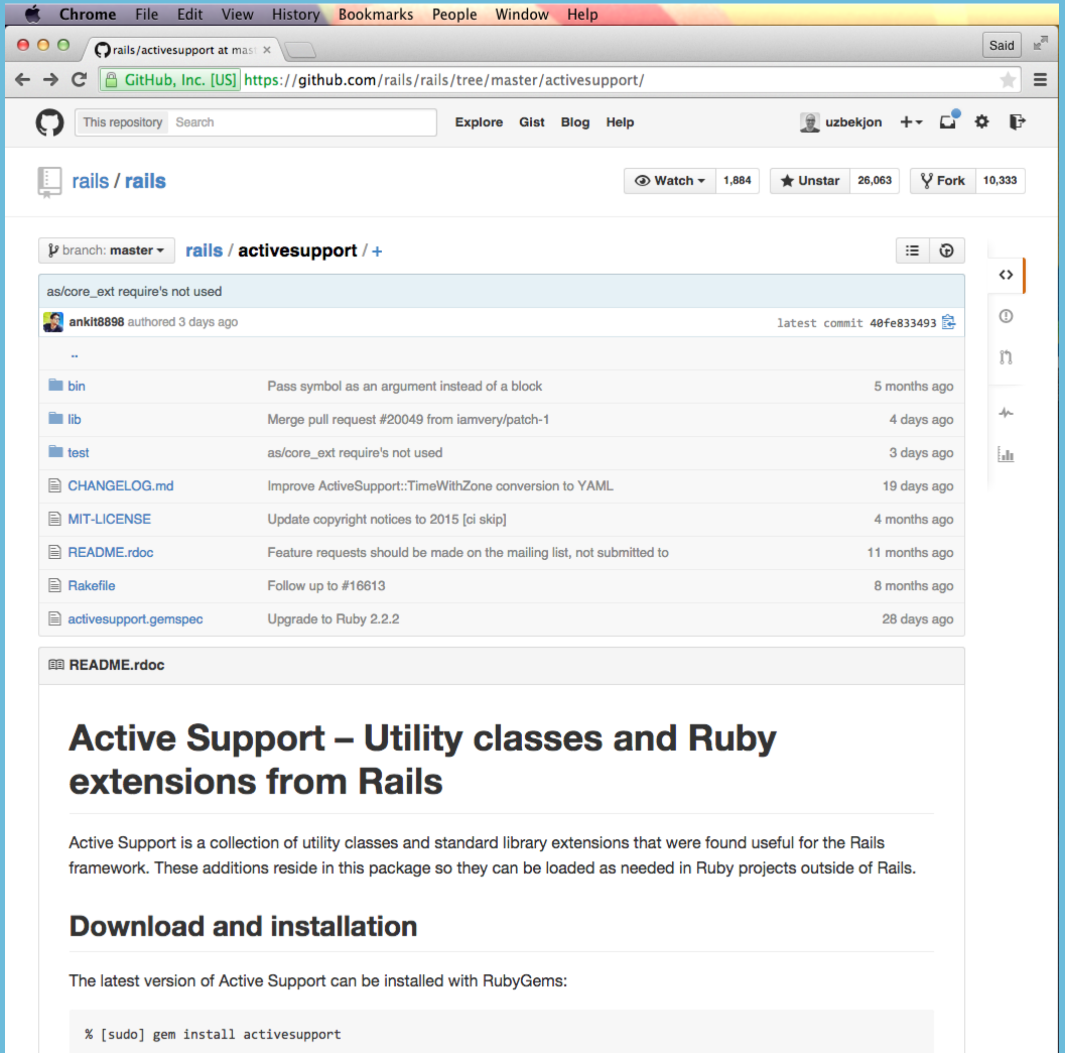
```ruby
class Module
    def alias_method_chain_imitator(original, extra)
        # Method body from previuos slide
    end
    private :alias_method_chain_imitator
end


class MyClass
    def foo
        "original foo"
    end

    def foo_with_extra
        "#{foo_without_extra} - foo with extra"
    end

    alias_method_chain_imitator :foo, :extra
end

MyClass.instance_methods false                              # [:foo, :foo_with_extra, :foo_without_extra]


obj = MyClass.new

obj.foo                                                     # "original foo - foo with extra"
obj.foo_with_extra                                          # "original foo - foo with extra"
obj.foo_without_extra                                       # "original foo"
```

Rails' "alias_method_chain" source code



```ruby
# Source code from:
#   https://github.com/rails/rails/blob/5bc77368/activesupport/lib/active_support/core_ext/module/aliasing.rb#L26
def alias_method_chain(target, feature)
    ActiveSupport::Deprecation.warn("alias_method_chain is deprecated. Please, use Module#prepend instead. From mc

    # Strip out punctuation on predicates, bang or writer methods since
    # e.g. target?_without_feature is not a valid method name.
    aliased_target, punctuation = target.to_s.sub(/([?!=])$/, ''), $1
    yield(aliased_target, punctuation) if block_given?

    with_method = "#{aliased_target}_with_#{feature}#{punctuation}"
    without_method = "#{aliased_target}_without_#{feature}#{punctuation}"

    alias_method without_method, target
    alias_method target, with_method

    case
    when public_method_defined?(without_method)
        public target
    when protected_method_defined?(without_method)
        protected target
    when private_method_defined?(without_method)
        private target
    end
end
```

Go through "ActiveSupport" source code


Chrome File Edit View History Bookmarks People Window Help
rails/activesupport at mas
Said
GitHub, Inc. [US] https://github.com/rails/rails/tree/master/activesupport/
This repository Search
Explore Gist Blog Help
uzbekjon
rails / rails
Watch 1,884
Unstar 26,063
Fork 10,333
branch: master
rails / activesupport / +

as/core_ext require's not used

ankit8898 authored 3 days ago latest commit 40fe833493

..		
bin	Pass symbol as an argument instead of a block	5 months ago
lib	Merge pull request #20049 from iamvery/patch-1	4 days ago
test	as/core_ext require's not used	3 days ago
CHANGELOG.md	Improve ActiveSupport::TimeWithZone conversion to YAML	19 days ago
MIT-LICENSE	Update copyright notices to 2015 [ci skip]	4 months ago
README.rdoc	Feature requests should be made on the mailing list, not submitted to	11 months ago
Rakefile	Follow up to #16613	8 months ago
activesupport.gemspec	Upgrade to Ruby 2.2.2	28 days ago

README.rdoc

Active Support – Utility classes and Ruby extensions from Rails

Active Support is a collection of utility classes and standard library extensions that were found useful for the Rails framework. These additions reside in this package so they can be loaded as needed in Ruby projects outside of Rails.

Download and installation

The latest version of Active Support can be installed with RubyGems:

```
% [sudo] gem install activesupport
```

Reading ActiveSupport source code is highly recommended!

Source:

https://github.com/rails/rails/tree/master/activesupport/lib